EMULEX VAX MONITOR (EVM)
USER'S GUIDE

3545 Harbor Boulevard
Costa Mesa, California 92626
(714) 662-5600 TWX 910-595-2521

VX9950901-00 Rev E
April, 1987

The information in this manual is for information purposes and is subject to change without notice.

Emulex Corporation assumes no responsibility for any errors which may appear in the manual.

Printed in U.S.A.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Section 1
GENERAL DESCRIPTION

1.1 Introduction

This manual is designed to serve as a guide for those using the Emulex VAX Monitor, EVM, on Digital Equipment Corporation (DEC) VAX-11 computers. EVM is used in conjunction with diagnostic programs for Emulex hardware in a VAX hardware environment.

These diagnostics are designed for use by qualified installers of Emulex equipment, and thus assume that the user has some knowledge of hardware configurations, VAX architecture and terminology, and interpretation of error messages and device register contents.

This document contains two main sections:

- **Section 1 General Description.** This section contains an overview of the Emulex VAX Monitor, including its functions, distribution media, hardware and software compatibility, and related documentation.
- **Section 2 Operation.** Describes operation of EVM, including bootstrap procedures, program control, monitor commands, and abbreviations.

1.2 Product Overview

The Emulex VAX Monitor (EVM) is a system program that provides an interface between the operator and the diagnostic programs. EVM provides the means by which diagnostic tasks are started, by which they communicate with the operator, and by which they are terminated. A diagnostic program is terminated by one of three methods:

- It can run to completion as requested.
- The user can cancel the program via the system console.
- The program can be halted by an error condition.

EVM resides in VAX memory with a diagnostic program, and it provides a framework for control and execution of diagnostic programs. Figure 1-1 depicts the relationship between the diagnostic programs and EVM. EVM also provides non-diagnostic, processor-independent services to the diagnostic programs. Most functions of EVM fall into one of two major areas:

- Command line interpreter
- Diagnostic program interface and control

The command line interpreter (CLI) consists of a command decoder and several service modules that execute the operator commands. Control is dispatched to the appropriate service module, depending upon the command. The program interface handles communication between the diagnostic program and EVM, and also provides program control services, message handling, error handling, terminal interfacing, and related functions.

System errors not related to the diagnostic programs are also handled by EVM, unless the program explicitly requests notification. The errors are transparent to the program and are reported directly to the operator.

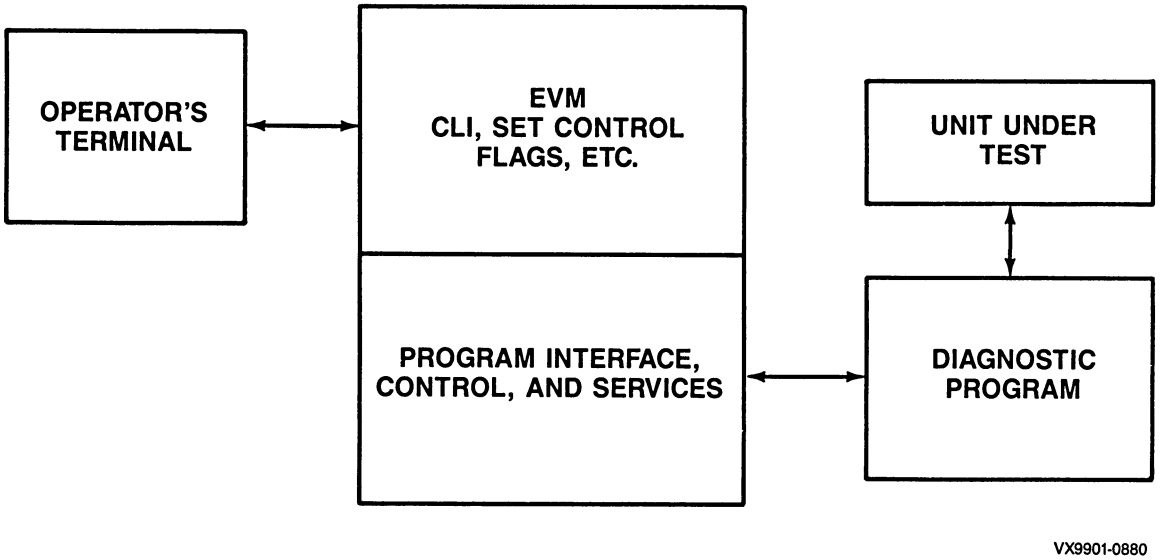

Figure 1-1. Monitor Environment

1.3 Distribution Media

The following table lists and describes distribution media for EVM.

Emulex P/N	Description
VX9960405-00	TU58 cassette for VAX-11/73x and VAX-11/75x
VX9960505-00	Eight-inch floppy diskette for VAX-11/78x

1.4 Compatibility

This section describes the hardware and software requirements of the Emulex VAX Monitor (EVM).

1.4.1 Hardware

EVM is compatible with DEC VAX 11/730, 11/750, and 11/780 computers.

1.4.2 Software

EVM is compatible with a number of Emulex diagnostic programs designed for use in a VAX environment. Documentation for these programs has been included in your software kit.

1.5 Related Documentation

If you require additional copies of this document, or of any Emulex diagnostic program user's guide, use the following address and telephone number for ordering:

Emulex Corporation
3545 Harbor Blvd.
Costa Mesa, CA 92626
(714) 662-5600 TWX 910-595-2521

Section 2
OPERATION

2.1 Overview

This section describes EVM bootstrapping procedures; interactions between the diagnostic monitor, the operator, and the diagnostic program; diagnostic monitor commands; and abbreviations for commands, switches, and literal arguments.

User input appears in **bold** type, in order to distinguish it from EVM or diagnostic program output. The names of function keys are enclosed in angle brackets; for example, the symbol **<return>** represents the RETURN or ENTER key. Both the symbol ^ and the abbreviation **ctrl** are used to represent the CONTROL key.

2.2 Bootstrapping

Insert the EVM diagnostic media into the appropriate console drive (normally CSA1:) and type one of the commands shown in this subsection.

For the VAX 11/730, enter the following:

```
>>> L/P/S:10000 DDx:EVM.EXE<return>
>>> S 10000<return>
```

NOTE

On the VAX 11/730, DD0: is the external TU58 drive and DD1: is the internal drive that holds the local console tape.

For the VAX 11/750, perform the following steps:

1. Locate the BOOT DEVICE switch on the front panel and set it to position A, which corresponds to the TU58 system device. This action matches the correct boot ROM with the VMS system device.
2. Set the POWER ON ACTION switch (located on the front panel) to the RESTART/BOOT position.
3. Press the RESET button.
4. The following message displays: %%

The VAX 11/750 can be bootstrapped using the local console if the boot block on TU58 becomes corrupted. The bootstrapping sequence is as follows:

1. Insert the local console TU58 in the console drive.

2. Bootstrap as follows:

   >>> B DDA0<return>
   Boot58>

3. Remove the local console TU58, insert the EVM diagnostic TU58, and, at the prompt, enter the following:

   Boot58>**LOAD EVM.EXE/START:10000<return>**
   Boot58>**START 10000<return>**

For the VAX 11/780, the command sequence is as follows:

>>> L EVM.EXE/S:10000<return>
>>> UNJAM<return>
>>> I<return>
>>> START 10000<return>

The following message and prompt then appear on the screen:

EMULEX VAX MONITOR REV n.m VAX-11/7XX
DD-MMM-YYY TIME
EVM>

Information regarding EVM commands can be obtained by typing HELP at the EVM> prompt, as also explained in subsection 2.4.14.

2.3 Monitor/Operator/Program Interaction

Once EVM has been loaded and started, program control moves to the initialization routine of EVM. This routine performs the following functions:

- Checks for 256K good bytes of memory
- Reads the processor ID
- Changes the processor mode to kernel
- Initializes various vectors, variables, and tables
- Sets the processor registers to a known state

It then initializes the system control block (SCB) and transfers control to the Command Line Interpreter (CLI) routine.

The CLI displays the prompt EVM>, indicating that EVM is ready for commands. The operator loads the diagnostic program, sets up various test control flags, and starts the diagnostic program. On receiving the start-program command, EVM clears the error count, sets the pass count to 0, and calls the program initialization routine.

The initialization routine may question the operator further concerning the unit to be tested or other parameters of the test. It also initializes the unit under test, sets up the conditions that are necessary to the diagnostic program, and returns control to the EVM dispatch routine.

EVM dispatches control to various test modules sequentially. If the TRACE flag is set, EVM prints the test title before calling each test routine. In turn, each test module initializes test-dependent variables and addresses and starts executing the code. Control is returned to EVM either at the end of the test or when an error condition occurs.

The EVM error handler prints out error messages, checks for program control flags, and takes action accordingly. At the end of the test, EVM checks to determine whether there is another test to be run. If so, EVM sequences control back to the first test selected.

Once the end of the pass has been reached, EVM determines whether or not the last pass to be run has been completed. If so, the cleanup and summary routines in the diagnostic program are called, and control returns to the command input routine. EVM types out the prompt EVM> and waits for another command from the operator. If the last pass has not been completed, EVM initiates the next pass of tests.

The operator can stop program execution at any time by entering **<ctrl-c>** at the console, as also explained in subsection 2.4.7.

2.4 Diagnostic Monitor Commands

EVM implements a set of commands and control flags that allow the operator to load and run the diagnostic programs and control program execution. The control flags are used to control the printing of error messages, ringing the bell, halting, and looping the program; they are program-independent and are therefore consistent across the range of diagnostic programs. Unless otherwise noted, commands are parsed and processed as far as possible, terminating when an error occurs.

The use of certain commands may differ slightly from one diagnostic program to another; for example, a few diagnostic programs do not yet include the HELP facility. In all such cases, the user's guide for the individual diagnostic program explains the exceptions.

Many of these commands refer to the diagnostic program currently loaded in memory, so they cannot be executed until a diagnostic has been loaded (using the LOAD command):

- SHOW CONFIGURATION
- SET CONFIGURATION
- PROGRESS
- CONTROL-C
- ABORT
- CONTINUE
- DEBUG
- START

Other EVM commands do not refer to a specific diagnostic program, so they may be entered before a diagnostic has been loaded:

- DIRECTORY
- SET TIME
- SHOW TIME
- SHOW FLAGS
- SET FLAGS
- CLEAR FLAGS

The information displayed by the HELP command depends upon whether or not a diagnostic program is currently loaded in memory. If no diagnostic is loaded, EVM displays information regarding monitor commands. If a diagnostic is loaded, the HELP command may be used with qualifiers to obtain information regarding that specific diagnostic program or some of the tests that it runs. For details regarding use of the HELP command, see subsection 2.4.14.

2.4.1 Command Syntax

Commands, switches, and literal arguments can be abbreviated to the minimum number of characters necessary to retain their unique identity. These minimum abbreviations are listed in subsection 2.5, Tables 2-2 and 2-3.

EVM commands use the following syntax:

EVM>COMMAND[/optional qualifier:<argument>]<return>

Command qualifiers, which are also called switches, and optional arguments are [bracketed]. Literal arguments, such as test numbers or filenames, are enclosed in <angle brackets>. Note that brackets and angle brackets are **not** part of the command and must not be typed by the operator.

A command qualifier is preceded by a slash (/), and is associated with a literal by a colon (:). A pair of literal arguments that specifies a range is separated by a colon; elements of an argument list are separated by commas. For example, the command statements

EVM>**LOAD IUT11<return>**
.
.
EVM>**START/TEST:1:5/PASSES:3<return>**

would load and begin execution of the diagnostic program currently in memory (in this example, the TC11 installation diagnostic). Tests 1 through 5 would run in three passes.

The following example illustrates the argument list format:

EVM>**SET FLAGS NOTRACE,BELL<return>**

As explained in subsection 2.4.9 and Table 2-1, this statement would prevent EVM from reporting execution of each test and would cause the bell to ring when an error message is printed.

2.4.2 LOAD Command

This command loads the specified file into memory from the console storage device (cassette or floppy only). The default file extension is .EXE. As explained in subsection 2.4, the diagnostic program must be loaded into memory using this command before program-specific commands can be used. Syntax:

EVM>**LOAD <filename><return>**

2.4.3 DIRECTORY Command

This command lists the files on the load device (cassette or floppy disk only). Syntax:

EVM>**DIRECTORY<return>**

Sample DIRECTORY output from EVM appears as follows.

FILE NAME	START BLOCK	# BLKS	CREATION DATE
EVM .EXE	14	161	15-OCT-84
FMD000.EXE	175	79	15-OCT-84
FUD12M.EXE	254	44	15-OCT-84
FUD31 .EXE	298	80	15-OCT-84
IVV000.EXE	378	6	15-OCT-84
< UNUSED >	384	110	

5 FILES, 370 BLOCKS
110 FREE BLOCKS
1 DIRECTORY SEGMENTS

2.4.4 SHOW CONFIGURATION Command

This command prints the hardware link of the unit under test, as defined by the diagnostic program and optionally updated by the SET CONFIGURATION command. Only those parameters appropriate to the device and the current VAX type are displayed. Syntax:

EVM>**SHOW CONFIGURATION<return>**

A display similar to the following example appears on the screen:

CONFIGURATION PARAMETERS FOR UNIT SELECTED ARE:

BR = 4
ADAPTER # = 1
CSR = 776700
VECTOR = 254
UBR = 5
DRIVE # = 0

2.4.5 SET CONFIGURATION Command

This command allows the operator to specify the hardware link to the unit being tested. Only those parameters for which the default values are incorrect need to be entered; these default values are displayed by the SHOW CONFIGURATION command. All parameters that are specified in a valid manner are set, regardless of whether any are erroneous.

After the entire command has been processed, if one or more required parameters remains undefined, an error message is printed which lists the mnemonics for the missing parameters. If the operator enters a parameter that is not appropriate to the device and the current CPU type, the command is accepted, but a warning message is displayed. The syntax of this command is as follows:

EVM>**SET CONFIGURATION<return>**
[/TR:<SBI TR # 1-15>]
[/BR:<adapter BR level 4-7>]
[/DRIVE:<drive # 0-7>]
[/CSR:<CSR address 760000-777776 octal>]
[/VECTOR:<UNIBUS vector 0-776 octal>]
[/UBR:<UNIBUS BR level 4-7>]
[/ADAPTER:<adapter # 0-3>]

The following examples illustrate the use of the SHOW CONFIGURATION and SET CONFIGURATION commands:

EVM>**SHOW CONFIGURATION<return>**

CONFIGURATION PARAMETERS FOR UNIT SELECTED ARE:

BR = 4
ADAPTER # = 1
CSR = 776700
VECTOR = 254
UBR = 5
DRIVE # = 0

EVM>**SET CONFIGURATION/ADAPTER:0<return>**
EVM>**SHOW CONFIGURATION<return>**

CONFIGURATION PARAMETERS FOR UNIT SELECTED ARE:

BR = 4
ADAPTER # = 0
CSR = 776700
VECTOR = 254
UBR = 5
DRIVE # = 0

Note that the value for ADAPTER here has changed to 0 as a result of the SET CONFIGURATION command.

2.4.6 SET FLAGS Command

This command results in the setting of the execution control flags specified by the argument list, **<arg-list>**. The argument list is a string of flag mnemonics from Table 2-1, separated by commas. All flags for which valid names are specified are set, regardless of the presence of an invalid flag name in the list. Syntax:

EVM>**SET FLAGS <arg-list><return>**

2.4.7 CLEAR FLAGS Command

This command results in the clearing of the flags specified by the argument list. See Table 2-1 for supported arguments. All flags for which valid names are specified are cleared, regardless of the presence of an invalid flag name in the list. Syntax:

EVM>**CLEAR FLAGS <arg-list><return>**

Table 2-1. Flag Mnemonics for SET FLAGS Command

FLAG	DESCRIPTION
HALT	Halt on error detection: When an error occurs, control returns to CLI to allow the operator to continue or abort as desired.
LOOP	Loop on error: This flag has no effect on EVM control of the diagnostic. It is testable by the diagnostic, which can loop on a given test if so specified.
BELL	Bell on error: If this flag is set, EVM rings a bell when an error message is printed.
QUICK	Quick test, program-dependent: This flag has no effect on EVM. It is testable by the diagnostic, which may take special action if desired.
IE	Inhibit error messages: If this flag is set, no error messages are printed.
NP	No print: If this flag is set, no messages are printed except prompt and CLI error messages.
NOTRACE	Do **not** report the execution of each test. If this flag is set, a message is not printed at the start of each test.
NO_OPERATOR	Operator **not** present. If this flag is set, default responses are used for operator input.
ALL	All flags in the list are set.

2.4.8 SHOW FLAGS Command

This command prints a list of the program execution control flags set by the operator. Syntax:

EVM>**SHOW FLAGS<return>**

2.4.9 START Command

This command causes EVM to begin diagnostic program execution.

Before initiating the program execution, EVM checks the validity of the hardware link table. The list of undefined parameters is printed; the operator must enter these parameters using the SET CONFIGURATION command, as explained in subsection 2.4.5.

The syntax of the START command is as follows. (Note that the command has been continued to a second line here because of space limitations.)

EVM>START[/TEST:<first>[:last]][/PASSES:<count>][/ERRORS:<max>] <return>

The [bracketed] expressions here are optional, as explained in subsection 2.4.1. In other words, you may simply enter **START** (or **ST**), or you may specify a range of tests to be performed.

- If **/TEST** is omitted, all tests are performed. If **/TEST** is included, then at least one test must be specified, as represented by **<first>** in the syntax statement. If a range of tests is indicated, the two are separated by a colon, again as shown in the syntax statement: **[:last]**. This item is enclosed in brackets because it would be omitted if only a single test were specified. If first and last arguments are defined, EVM passes control to the tests in the range specified (first through last).

NOTE

For a few EVM compatible diagnostic programs, the rules governing use of the /TEST qualifier with the START command differ from the general procedure described here. In all such cases, the correct use of /TEST is explained in the appropriate diagnostic program user's guide.

- Another option, **/PASSES**, specifies the number of passes for each test. If no number is specified, the default value is 1. This option can be used with or without the **/TEST** option, because if no test switch is specified, all tests of the program are executed for the specified number of passes. Note that 0 here signifies an infinite number of passes, rather than zero passes.

- A third option, **/ERRORS**, allows the operator to specify the maximum number of errors which EVM will accept. If that number is reached, program execution is stopped and that fact is reported to the operator via the terminal. The operator may then either abort or continue (see subsections 2.4.8, ABORT command, and 2.4.12, CONTINUE command).

The following example illustrates the use of the LOAD, SET CONFIGURATION (subsection 2.4.5), and START commands for a VAX-11/780 with one Emulex CS21/F1 communications multiplexer configured as follows:

- CSR address 760340 octal
- Interrupt vector address 300 octal
- UNIBUS address UBA1:
- UNIBUS BR level 4

UBA1: corresponds to TR 4, as explained in the following text. IUC32F is the name of a diagnostic program, for which the operator has requested three passes:

```
EVM>LOAD IUC32F.EXE<return>
EVM>SET CONFIG/CSR:760340/VECTOR:300/TR:4/BR:4<return>
EVM>START/PASSES:3<return>
```

The qualifier /TR needs to be specified only if the TR (transfer request) number is other than 3 (UBA0:). Acceptable values for this parameter are 3 through 6, with 3 the default:

TR 3	UBA0, 20100000
TR 4	UBA1, 20140000
TR 5	UBA2, 20180000
TR 6	UBA3, 201C0000

/BR needs to be specified only if the device BR (bus request) level is other than 5. Acceptable values for this parameter are 4 through 7, with 5 the default.

As previously explained, /PASSES needs to be specified only if multiple passes are desired; the default value is 1. Acceptable values for this parameter are 0 through 100, where 0 signifies an infinite number of passes.

Figure 2-1 illustrates sample VAX diagnostic program output, in this case for the Emulex CS21/CS11/CS32 F1 installation diagnostic (IUC32F). For an explanation of this program, see the IUC32F user's guide referenced in subsection 1.5 of this document.

After the test has run to completion, control is returned to EVM.

```
Emulex VAX-UNIBUS DMF diagnostic software  REV 1.0  DD-MMM-YYYY TIME
Do you wish to run with loopback connectors? [ (N) ]>>> N<return>
Number of emulations:  [DEC - 1,32,(1)]>>> 2<return>
    Enter line mask for emulation 0
                        [HEX - 0,FF,(FF)]>>> FF<return>
    Enter line mask for emulation 1
                        [HEX - 0,FF,(FF)]>>> FF<return>

    TEST # 1 Master reset & register addressability  dd-mmm-yyyy
    TEST # 2 TX ready, TX enable and TX silo test  dd-mmm-yyyy Time
    TEST # 3 TX enable/disable  dd-mmm-yyyy Time
    TEST # 4 Character length test  dd-mmm-yyyy Time
    TEST # 5 Stop bit test  dd-mmm-yyyy Time
    TEST # 6 Parity bit test  dd-mmm-yyyy Time
    TEST # 7 Load word test  dd-mmm-yyyy Time
    TEST # 8 Flush silo test  dd-mmm-yyyy Time
    TEST # 9 Preempt test  dd-mmm-yyyy Time
    TEST # 10 TX break test  dd-mmm-yyyy Time
    TEST # 11 TX interrupt test  dd-mmm-yyyy Time
    TEST # 12 RX interrupt test  dd-mmm-yyyy Time
    TEST # 13 Multiple interrupt  dd-mmm-yyyy Time
    TEST # 14 DMA data transfer  dd-mmm-yyyy Time
    TEST # 15 NPR-Nonexistent memory error  dd-mmm-yyyy Time
    TEST # 16 Odd and even address boundary test  dd-mmm-yyyy Time
    TEST # 17 DMA with memory extension test  dd-mmm-yyyy Time
    TEST # 18 Data transfer external loopback (wrap around)
    No wrap around connector, test skipped
    TEST # 19 Split baud rate (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 20 RX & TX modem signals (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 21 Auto echo mode (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 22 Data reliability test (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 23 Dynamic baud rate (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 24 Dynamic word length test (wrap around)  dd-mmm-yyyy
    No wrap around connector, test skipped
    TEST # 25 Dynamic stop bit test (wrap around) dd-mmm-yyyy Time
    No wrap around connector, test skipped
    TEST # 26 Dynamic parity test (wrap around)  dd-mmm-yyyy Time
    No wrap around connector, test skipped

    SUMMARY REPORT:
    TOTAL # ERRORS = 0 (0 SYSTEM, 0 DEVICE, 0 HARD, 0 SOFT)
    dd-mmm-yyyy Time

    EVM>
```

Figure 2-1. Sample Diagnostic Program Output

2.4.10 PROGRESS Command

This command causes the program to print a progress report. EVM generates a header containing the following information:

- The name of the diagnostic program
- The current pass number and test numbers
- A summary of hard, system, soft, and device errors
- Diagnostic-specific progress data

The syntax for this command is:

PROGRESS<return>

2.4.11 CONTROL-C Command

When the operator types **<ctrl-c>** (that is, the CONTROL key and the C character key pressed simultaneously), the program is not interrupted immediately. The command merely sets a flag, and the status of this flag is checked periodically by EVM. If the flag is set, the CLI is called. The operator may then issue any of a number of commands to continue or abort the test.

EVM gets control, as indicated by the EVM> prompt on the screen, and waits for the operator's command. If the diagnostic program is running, the operator may issue any command except LOAD, START, or SET CONFIGURATION. Syntax:

^C

2.4.12 CONTINUE Command

This command resumes program execution at the point at which the program was suspended by **<ctrl-c>**. This command is invalid if no tests are in progress (that is, if no test was selected, or following an abort). Syntax:

EVM>**CONTINUE<return>**

2.4.13 ABORT Command

This command passes control to the program's cleanup code, re-initializes the system, and then returns control to EVM, which enters the Command Wait state and displays the prompt EVM>. Any command except CONTINUE or PROGRESS may be issued at this time. Syntax:

ABORT<return>

2.4.14 HELP Command

This command prints help information. Two optional qualifiers are available, /DIAG and /TEST, which can be used only while a diagnostic program is loaded in memory.

- If **no** qualifier is specified, EVM prints this message, followed by help information regarding EVM monitor commands:

  FOLLOWING EVM COMMANDS ARE AVAILABLE

- If the **/DIAG** qualifier is used, EVM prints a description of the diagnostic program currently in memory. Syntax:

  EVM>**HELP/DIAG<return>**

- If the **/TEST** qualifier is used, EVM prints a description of tests available with the diagnostic program currently in memory. If you do not specify any test numbers, a description of all available tests is printed. To request descriptions of a subset of diagnostic tests, use the following command syntax:

  EVM>**HELP/TEST:<first>:[<last>]<return>**

NOTE

A few EVM compatible diagnostic programs do not yet include the HELP facility. In all such cases, the user's guide for the individual diagnostic program explains the exceptions and describes all available tests.

2.4.15 DEBUG Command

This command is used to invoke EVM's debug facility.

NOTE

The DEBUG command is provided as an aid in diagnostic program development and modification, and is not intended for customer use.

For help on debug commands, type **HELP** at the DBG> prompt. The following text will be displayed:

```
EVM>DEBUG<return>
DBG>HELP<return>

AVAILABLE COMMANDS ARE:
    CANCEL BREAK/ALL
    CANCEL BREAK address
    DEFINE symbol = expression
    DEPOSIT address = expression[,expression,...]
    DEPOSIT NEXT:n address = expression
    DEPOSIT address = "string"
    DEPOSIT/NEXT:n address = expression
    DUMP address:address
    DUMP/NEXT:n address
    EXAMINE address[,address,...]
    EXAMINE address:address[,address:address,address...]
    EXAMINE/NEXT:n address
    EXIT
    EVALUATE expression
    EVALUATE/ADDRESS address
    GO [address]
    MODIFY address
    SET BREAK[AFTER:n] address
    SET MODE radix
    SET TYPE length
    SHOW BREAK
    SHOW MODE
    SHOW SYMBOL
    SHOW TYPE
    STEP [n]

DBG>
```

2.4.16 SET TIME Command

This command is used to set the system time. Syntax:

EVM>**SET TIME dd-mmm-yyyy hh:mm:ss<return>**

2.4.17 SHOW TIME Command

This command is used to display the system time. Syntax:

EVM>**SHOW TIME<return>**

2.4.18 SET OPTIONS Command

This command results in the setting of various options. (In this version of EVM, there is only one option: UNSOLICITED.) Syntax:

EVM>**SET OPTIONS/UNSOLICITED=nn<return>**

The system prints the messages for the first "nn" occurrences of unsolicited interrupts and stops printing messges after that. EVM has a default value of UNSOLICITED=5 when first loaded.

2.4.19 SHOW OPTIONS COMMAND

This command prints the OPTION flags set by the operator using the SET OPTIONS command. Syntax:

EVM>**SHOW OPTIONS<return>**

2.5 Abbreviations

Commands, switches, and literal arguments can be abbreviated as shown in Tables 2-2 and 2-3. The characters in **bold** type represent the minimum abbreviations needed; characters in [brackets] can be omitted. Each qualifer begins on a new line of text here because of space limitations.

Table 2-2. Command Abbreviations

Command[1]	Subsection
A[BORT]	2.4.13
CL[EAR] **F**[LAGS] <arg-list>	2.4.7
CO[NTINUE]	2.4.12
CONTROL C	2.4.11
DE[BUG]	2.4.15
DI[RECTORY]	2.4.3
H[ELP][/**D**[IAG]][/**T**[EST]:<first>[:last]]	2.4.14
L[OAD] <filename>	2.4.2
P[ROGRESS]	2.4.10
SE[T] **C**[ONFIGURATION] [/**T**[R]:<SBI TR# 1-15>] [/**B**[R]:<adapter BR level 4-7>] [/**DR**[IVE]:<drive # 0-7>] [/**C**[SR]:<CSR address 760000-777776 octal>] [/**V**[ECTOR]:<UNIBUS vector 0-776 octal>] [/**U**[BR]:<UNIBUS BR level 4-7>] [/**A**[DAPTER]:<adapter #0-3>]	2.4.5
SE[T] **F**[LAGS] <arg-list>	2.4.6
SE[T] **O**[PTIONS][/UNS[OLICITED=<max>]	2.4.18
SE[T] **T**[IME] dd-mmm-yyyy hh:mm:ss	2.4.16
SH[OW] **C**[ONFIGURATION]	2.4.4
SH[OW] **F**[LAGS]	2.4.8
SH[OW] **O**[PTIONS]	2.4.19
SH[OW] **T**[IME]	2.4.17
ST[ART] [/**T**[EST]:<first>[:last]] [/**P**[ASSES]:<count>] [/**E**[RRORS]:<max>]	2.4.9

1. Minimum abbreviations appear in **bold** type.

Table 2-3. Flag Abbreviations

Flag[1]	Definition
A[LL]	All flags in the list
B[ELL]	Bell on error
H[ALT]	Halt on error detection
IE	Inhibit error messages
NP	Do not print any messages
L[OOP]	Loop on error
NO_O[PERATOR]	Operator not present
Q[UICK]	Quick test, program dependent
NOT[RACE]	Do not report the execution of each test

[1]Minimum abbreviations appear in **bold** type. See also subsection 2.4.6, SET FLAGS command.

BLANK